M.I.T. Media Laboratory Vision and Modeling Group Technical Report No. 314

Second-order method for occlusion relationships in motion layers

Trevor Darrell and David Fleet†
trevor@media.mit.edu fleet@qucis.queensu.ca

ABSTRACT

A new method is presented to recover the depth ordering of motion layers in a scene. We derive a measure for determining the occlusion relationship between two layers, by testing whether the support of one motion layer is moving with the predicted local velocity of another motion layer. This is used to construct a graph which represents the depth ordering of all layers. By cascading two first-order motion models, a second-order model is defined which is sensitive to the motion of motion-defined support regions, which is sometimes called kinetic occlusion. Our algorithm makes few assumptions about the motion and layer models, and can use any motion layer model which provides global hypotheses in the form of velocity fields, and any local motion mechanism which provides the conditional probabilty of a velocity given a local image region.

1 Introduction

The problem of how to segment and group information about visual motion is one of the most difficult issues in computational models of motion perception. Due to ambiguities inherent in the motion measurment process, information at a single point is insufficient to resolve the true underlying motion in the scene at that point. To overcome this well-known *aperture problem*, information about motion must be integrated across multiple points in the scene. Models of motion perception must therefore both estimate motion parameters (local velocity, affine motion, etc..) *and* determine the segmentation of the scene to recover valid information.

Early approaches to this problem attempted to find the boundaries of homogeneous motions in the scene and estimate parameters within regions defined by these edges. This has been formalized as a *line-process* within the framework of Markov Random Fields[16], or with the notion of controlled disconinuities within a regularization framework[5, 18]. More recently, the concept of *layers* was introduced to provide a more general representation for scene segmentations, and to facilitate a richer description of images in terms of their intrinsic properties [1].

Several authors have implemented methods that compute motion layers. Darrell and Pentland[8] applied a model of cooperative robust estimation to the motion domain, decomposing an image sequence into a set of *support layers* that represent homogeneous motions (parameterized by a global model). Irani and Peleg[17] and Black[3] used higher-order motion models, such as the affine model, to segment moving objects and find regions of support. Wang and Adelson[20] used an affine model and a k-means clustering method to find motion layers, from which extended representations were constructed by compositing layers from several consecutive frames of an image sequence into a larger image. These authors have found that layers are an effective representation for problems which are not well suited for edge-based models, such as those involving transparency or disjoint occluded features. Layered representations draw from an establisted tradition of robust estimation methods[7], which are of increasing interest in the computer vision community[12, 13].

Ideally, a layered representation might also produce, or facilitate the inference of, a partial ordering of the layers that corresponds to a depth ordering of the surfaces of the scene. Inference of depth ordering has been achieved in layered representations of static images based on edges, with the assumption that T-junctions indicate occlusions [14]. However, the published methods for recovering layers based on motion information have not extracted cues for depth ordering. It may be possible to find T-junctions in space-time which would imply occlusion boundaries [2], but their detection in space-time remains a formidable problem.

In this paper we present a new approach for recovering occlusion relationships between layers, using a motion-based analysis that does not rely on static scene cues. This *second-order* method is a cascade of two mechanisms for computing conventional motion-layer support, and it provides direct evidence for the occlusion relationship of the underlying surfaces. In effect, if the motion of a boundary of a support region itself is consistent with the motion model of the intensity structure within that region, then the region can be said to *own* that boundary. Conversely, if the motion of a support region is consistent with the motion model of a different layer, than this is direct evidence that the other layer is

†David Fleet is with the Robotics and Perception Lab, Department of Computing Science, Queen's University

occluding the layer associated with that region of support.

As we will show in the following sections, a second-order mechanism is straightforward to define from existing motion layer models, fits into a developed theory of higher-order motion methods, and can be implemented efficiently upon any method for finding first-order layers.

2 Models of motion layers

Motion layers are usually defined by regions of coherently moving image structure. Methods for recovering layers typically combine a local mechanism for extracting local constraints on optical flow, with a gobal mechanism for constraint selection or clustering. The global constraint is often a parametric model of flow[17, 20, 3, 8], but could be defined by a smoothness constraint, such as regularized optic flow. The constraint for a particular layer can be expressed as a global velocity field, depicting how points in the image would move if such motion were present in that region of the scene.

We view a layered representation as a decomposition of an image into a set of global hypotheses, each of which is represented as a velocity field. We restrict our attention to instantaneous layers that describe the image information at a single moment in time. Methods which composite information from multiple frames into an extended layer have been developed[20], and our framework here can be generalized naturally to cover that case.

Formally, a layered representation $\mathcal{L}$ provides a decomposition of an image sequence $I(x,y,t)$ into a set of layers l_k with velocity fields $v_k(x,y)$ that represent homogeneous (globally consistent) motion constraints, and support maps $s_k(x,y)$ that express where those constraints agree with the observed image intensities at $t=0$. Regions of support need not be connected within any given layer, and if there is transparency in the scene, then more than one motion layer may be supported at a single image location[9]. We define

$$\begin{aligned}\mathcal{L} &= \{l_0,\ l_1,\ \ldots,l_K\}\\ &= \{(v_0(x,y),s_0(x,y)),\ (v_1(x,y),s_1(x,y)),\\ &\quad \ldots,\ (v_K(x,y),s_K(x,y))\}\end{aligned} \tag{1}$$

The support for a motion layer is defined to be near 1 where the constraint is valid, and near 0 where the constraint is invalid. It can be viewed as an *inverse-outlier* measure when used in robust estimation methods[7, 4]. Thus

$$s_k(x,y) = C_{x,y,0}(I, v_k(x,y)), \tag{2}$$

where $C_{x,y,t}(I,v)$ is a constraint function which effectively supports whether velocity v is present in the image sequence I at time t and location (x,y). If we have a probabalistic motion model than a natural expression for the contraint function is the conditional probability of v given the observed intensities I.

$$C_{x,y,t}(I,v) = P(v|I_{x,y,t}) \tag{3}$$

If we consider sequences without additive transparency we can use the model developed by Simoncelli [19] which has a particularly simple form:

$$P(v|I_{x,y,t}) \approx \exp^{-\frac{(D(v)I_{x,y,t})^2}{||\nabla I_{x,y,t}||^2}} \tag{4}$$

where

$$D(v) = \begin{bmatrix} v_x \\ v_y \\ 1 \end{bmatrix} \cdot \nabla = (v_x\frac{\partial}{\partial x} + v_y\frac{\partial}{\partial y} + \frac{\partial}{\partial t})\ . \tag{5}$$

and $I_{x,y,t}$ is the input sequence offset by (x,y,t). When additive transparency occurs, higher-order filters must be applied to test motion support [9].

$C_t(I,v)$ provides a method to test the support for a local velocity that is implied by a particular layer. A method for recovering a layered description typically combines this type of test with a mechanism for determining the set of velocity fields $v_k(x,y)$ that is an optimal representation of the image. This has been accomplished using clustering methods, based either on the k-means algorithm[20, 3], or on a Minimum Description Length principle to select from a set of hypotheses[7, 11, 15].

These detailed methods are beyond the scope of this paper. Indeed, the method presented here can be applied with any model of layer recovery that returns a set of layers $\mathcal{L}$ as defined above. (The examples shown in this paper were obtained using an implementation of the layer recovery method given in [8].)

3 Scene ambiguity and boundary ownership

Given a layered representation of an image sequence, we want to determine a depth ordering of objects in the scene that correspond to the layers. We want to be able to make inferences about the physical structure of the scene, and detect when one object (or more generally, process) is occluding another. Since no assumptions are made concerning the static intensities of objects in the scene, we will rely only on motion cues.

The key idea is to use the information about the motion layers at several neighboring times, so that the motion of the support boundary itself can help disambiguate the depth ordering of the regions. Our approach to finding layer ordering is to devise a method for determining the *ownership* of support boundaries between different motion layers. We then determine the ordering relationship of two layers based on the ownership attributes of the boundary between them.

We can see the ambiguity of motion layers at a single point in time with a simple example: Given two ajoining layers, like those in Figure 1 with l_1 on the left and l_2 on the right, there are several possible interpretations of the physical ordering of surfaces that might have generated the sequence. In one case, the background of the scene is moving with

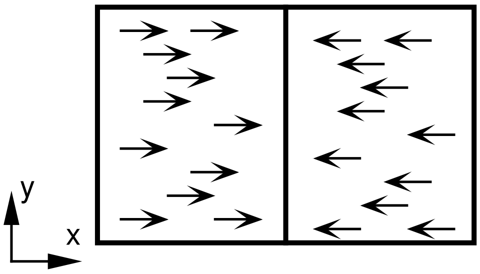


Figure 1: Two regions (l_1 on the left and l_2 on the right) are defined by coherent motions at a single moment in time. A motion-defined boundary exists between the two regions, but with no static cues the depth ordering of the regions is ambiguous.

velocity v_1, and a foreground object with shape s_2 is moving with velocity v_2. Conversely, the background could be moving with velocity v_2 and the object with velocity v_1. A third possibility of course is that there is no occlusion, which might occur, for example, because the boundary is actually an edge that is shared by two adjacent surfaces, or because two surfaces are adjacent and moving independently (as with neighboring conveyor belts). With motion information at a single instant, we can not decide among these hypotheses.

However if we have motion information at multiple moments in time, the ordering of layers can be disambiguated. The key observation is that the motion of the *aperature* (or boundary) of the moving region is a cue to the occlusion relationship. Given the images shown in Figure 2 we can see that the boundary of the leftward moving region is itself moving to the left as time progresses. This leads to a strong perception in human observers of a depth relationship in which the leftward moving region is in front of, and is occluding, the rightward moving region.

When the motion of an boundary agrees with the motion of the underlying intensity structure, then we can infer the boundaries of that aperature are associated with the moving intensity structure and may be occluding neighbouring regions. Situations can occur in the real world in which an aperature moves with motion different than an underlying region, such as motion viewed in reflection through a stationary aperature; these scenes are understood without difficulty. However the alignment of motion of both an aperature and a separate texture viewed in reflection through that aperature occurs quite rarely. In this case when there are no static cues available, such as the display of random dot motion on a computer screen, the agreement of motion of aperature and pattern is interpreted by human observers as being due to the motion of a single, unoccluded, moving object.

4 Boundary ownership from second-order motion

Following Cavanagh [6], first-order motion refers to the motion of image structure, extracted using constraints applied to the original image or a band-pass filtered version of it. Second-order motion, by comparison, refers to the motion of first-order properties such as regions of uniform texture, or regions of consistent motion. In short, second-order motion refers to the motion of regions defined by uniform (or coherent) first-order properties. Second-order motion has also been refered to as non-Fourier motion in the psychophysics literature, and related to a number of distinct phenomena, including the motion of contrast envelopes, subsampled time-varying signals, and of course the motion of occlusion boundaries [10]. In the current context, we are interested in the motion of regions of coherent motion, that is, the motion of the aperture (or occlusion) boundaries. In these terms, the original methods for recovering motion-layers outlined above are first-order mechanisms, because they extract motion primarily from constraints applied to the image intensity structure.

To measure the motion of an aperature defined by a region with a particular parametrized global motion, we can construct a second-order mechanism which is a cascade of two first-order mechanisms. To compute the support maps for each layer we are interested in testing the support for a hypothesized velocity, rather than estimating a velocity in general. For example, as shown in [19], the first-order stage can return a velocity-selective measure, which could be implemented, for example, using spatiotemporal linear filters to compute a gradient constraint, followed by a squaring and a normalization stage.

Our method for computing boundary ownership involves computing the support of a hypothesis that the region of support in a given layer is moving with a particular velocity. That is, we apply the support operator recursively, on the temporal sequence of

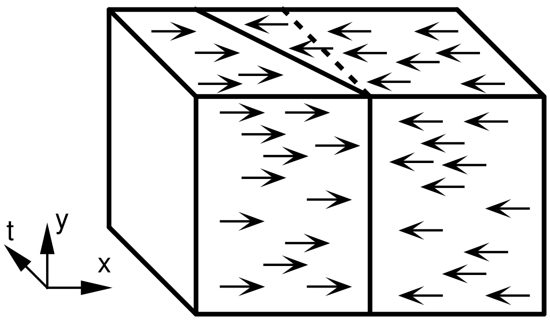


Figure 2: Two regions regions of coherent motion, as in the previous figure, but with regions of support computed over several frames, yielding a spatio-temporal volume of support for each layer. The motion of the support boundaries disambiguates the layer-ordering of the scene.

the support maps. Recall that $C_{x,y,t}(I,v)$ defines the support for velocity v at location x, y and time t in image sequence I. We similarly define second-order support (at $t = 0$) for a motion layer to be:

$$\begin{aligned} ss_k(x,y) &= C_{x,y,0}(s_k(x,y,t), v_k(x,y)) \\ &= C_{x,y,0}(C_{x,y,t}(I, v_k(x,y)), v_k(x,y)) \end{aligned} \tag{6}$$

Regions of the scene where both $s_k(x,y)$ and $ss_k(x,y)$ are large are defined to be boundary regions which are *owned* by layer l_k.

Figure 4 shows the first and second-order support for an image sequence constructed as in Figure 2. From the first-order support, we cannot determine boundary ownership. From the second-order support, the occluding layer has a strong response at the boundary, while the occluded layer has little or no response. We can therefore infer that the leftward moving layer *owns* the boundary between the two layers. Figure 5 shows another version of the first and second-order support maps computed for an image sequence with two motion layers, corresponding to a moving person and a stationary plant.

5 Layer ordering from boundary ownership

To determine ordering, we need to determine the occlusion relationships present in a set of layers. Boundaries of an object which are in front of (and thus are occluding) other regions will be "owned" by the layer corresponding to that object that is closer to the viewer. However, the converse is not necessarily true. Ownership of a boundary does not strictly imply occlusion at that boundary, at least in motion boundaries that can be realistically estimated from an image signal.

In all but the most uniform patterns, contrast variation induces many suprious boundaries in the support of a moving region. These spurious boundaries will be unoccluded and will move with the underlying motion, so they would indeed be owned by the layer associated with the motion of that region. However it would be incorrect to interpret these as occluding boundaries, since they are not actual (e.g. physical) boundaries of the object.

In addition, when we consider the case of transparent motion layers, it becomes clear that boundary ownership does not always imply occlusion. If we have two transparent moving surfaces, a layered decomposition of them would produce two support maps, which would each move over time according to the motion of each image. Indeed, by definition, each would have ownership of all boundaries in their support. Yet there would be no meaninful occlusion relationship between them because they are transparently combined.

The key insight to determining occlusion relationships from boundary ownership is to pose the negative question, that of finding boundaries of a layer that are not owned by that layer, and are in fact owned by *another* layer. Occlusion is implied when the boundaries (support) of the one layer are moving with the motion implied by another layer. We define an occlusion relationship $l_i <<< l_j$ to hold over regions of x, y where we can find a pair of layers $(v_i(x,y), s_i(x,y))$,$(v_j(x,y), s_j(x,y))$ such that $s_j(x,y)$ is moving with velocity $v_i(x,y)$. Our occlusion measure is thus

$$\begin{aligned} o_{ij}(x,y) &= C_{x,y,0}(s_j(x,y,t), v_i(x,y)) \\ &= C_{x,y,0}(C_{x,y,t}(I, v_j(x,y)), v_i(x,y)) \end{aligned} \tag{7}$$

and we say $l_i <<< l_j$ ("l_i occludes l_j") when $o_{ij}(x,y)$ is sufficiently large.

6 Algorithm for layer ordering

We have implemented an algorithm for determining layer ordering using the "occlusion maps" defined

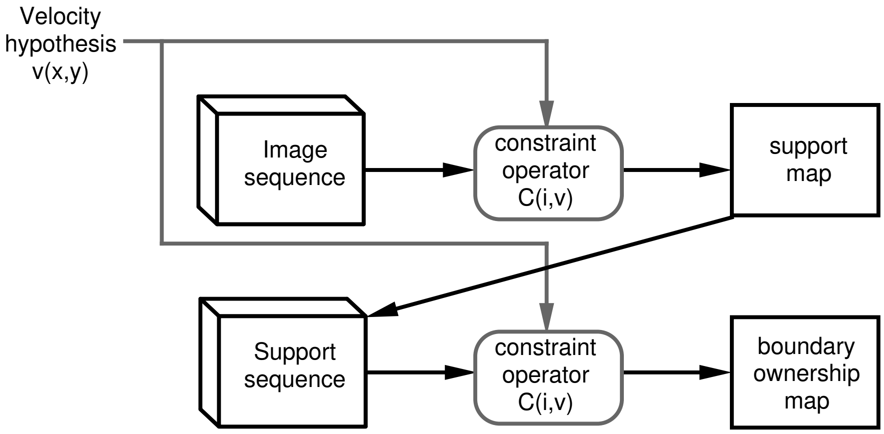


Figure 3: Construction of second-order motion operator. Second-order motion support can be computed by chaining two first-order support operators.

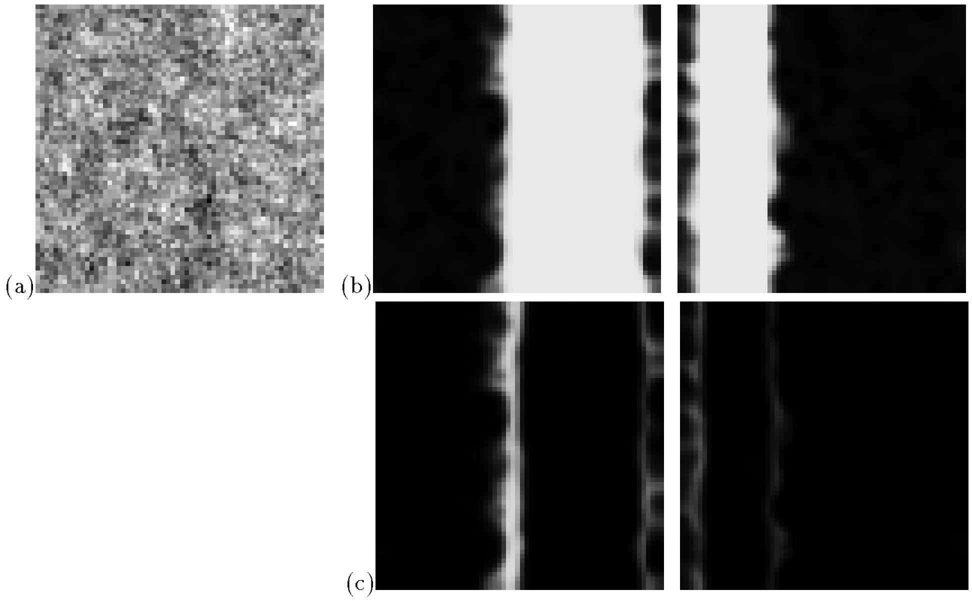


Figure 4: Motion support layers for an image sequence constructed as in Figure 2 using regions textured with random noise patterns. (a) Frame from sequence. No static segmentation cues are present; segmentation into layers and determination of occlusion/ordering is based purely on motion cues. (b) First-order support layers $s_0(x,y)$,$s_1(x,y)$ for this sequence ($v_0(x,y) = [-1,0]^T$, $v_1(x,y) = [1,0]^T$). (c) Second-order support layers $ss_0(x,y)$,$ss_1(x,y)$ for this sequence. The presence of second order support indicates boundary ownership, i.e., regions where that layer is not occluded.

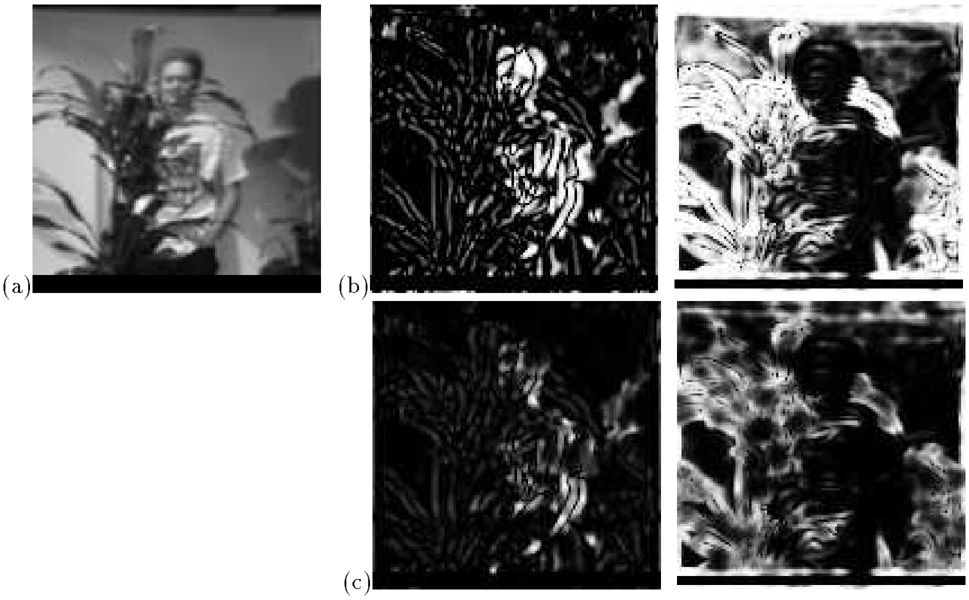


Figure 5: Motion support layers for an image sequence of a person (and shadow) moving behind a plant. (a) Frame from sequence. (b) First-order support layers $s_0(x,y)$, $s_1(x,y)$ ($v_0(x,y) = [-0.3, 0]^T$, $v_1(x,y) = [0,0]^T$).. (c) Second-order support layers $ss_0(x,y)$, $ss_1(x,y)$.

in the previous section. Our method begins with a set of first-order layers as input, computes occlusion maps for each pair of layers, and returns a directed graph indicating the ordering relationship between the layers determined from second-order motion cues.

We use the following algorithm: *Given* a set of first-order layers $\mathcal{L}$ and a constraint operator $C(I, v)$,

1. Compute spatio-temporal support for each layer, $s_k(x,y,t)$.

2. For each pair of layers i, j, compute the cross-layer occlusion map $o_{ij}(x,y)$

3. Sum over occlusion maps to compute an occlusion matrix $\mathbf{O}$

$$O_{ij} = \sum_{x,y} o_{ij}(x,y)$$

4. Construct directed graph representing layer ordering from the occlusion matrix. Let $\mathcal{O}$ be a directed graph with one node per layer, and a link from node i to node j when O_{ij} is large.

And then *return* the layer ordering graph $\mathcal{O}$, and the occlusion maps $\{o_{ij}(x,y), 0 < i,j < K\}$.

If $\mathcal{O}$ can be expressed as a singlely linked list, then we have the cannonical case of layers that are strictly ordered in depth (Figure 6(top)). In this case the occlusion maps $o_{ij}(x,y)$ are redundant, since all the information about ordering from second-order motion cues is present in the ordering graph. If $\mathcal{O}$ is a tree (Figure 6(middle)) then there are two or more layers whose relative depth is undefined from second-order motion cues, and only a partial ordering is available. If $\mathcal{O}$ has cyclic structure (Figure 6(bottom)), then there are layers which are not strictly ordered in the scene, and which interpenetrate or overlap in opposite ways depending on the particular location of the scene. In this case, the occlusion maps $o_{ij}(x,y)$,$o_{ji}(x,y)$ show the location dependent ordering of the two layers l_i, l_j .

Figures 7-10 illustrate the recovery of a layer ordering graph, for two image sequences with similar first-order motion patterns but different occlusion relations and thus different ordering graphs.

7 Conclusion

We have presented a new method which uses second-order motion information to recover the depth ordering of motion layers. Mechanisms for second-order motion can be simply expressed as the cascade of two first-order velocity selective mechanisms, separated by a pointwise non-linearity[10]. We combine this model with the layers approach to motion segmentation[20, 8], and observe that second-order motion in layers is an important cue to occlusion relationships and depth ordering.

One advantage of the layers model is that global constraints on the velocity field allow the local measurement mechanism to use velocity hypothesis testing rather than velocity estimation. This can be much more robust, especially when higher-order

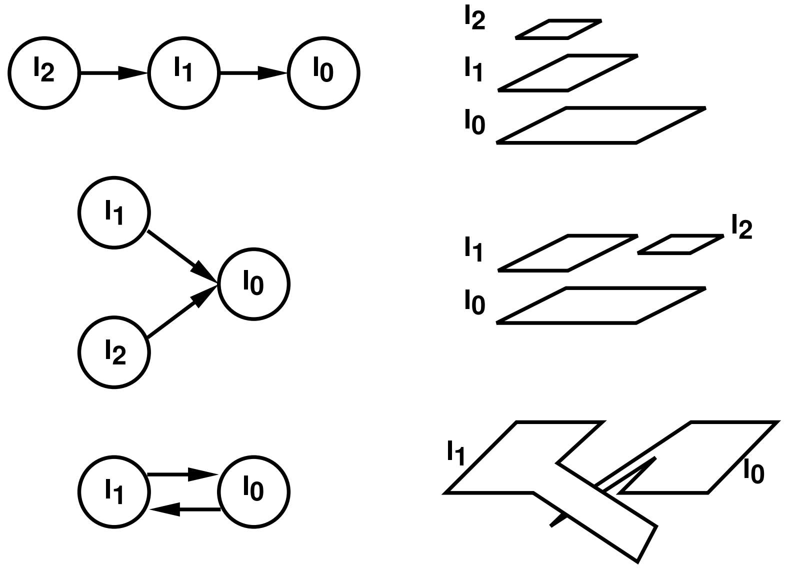


Figure 6: Layer ordering graphs and their equivalent physical realization. (top) A simple list corresponding to conventional depth layers. (middle) A tree corresponds to layers whose relative depth is unknown. (bottom) Cyclic structures indicate dual interpenetration or overlap.

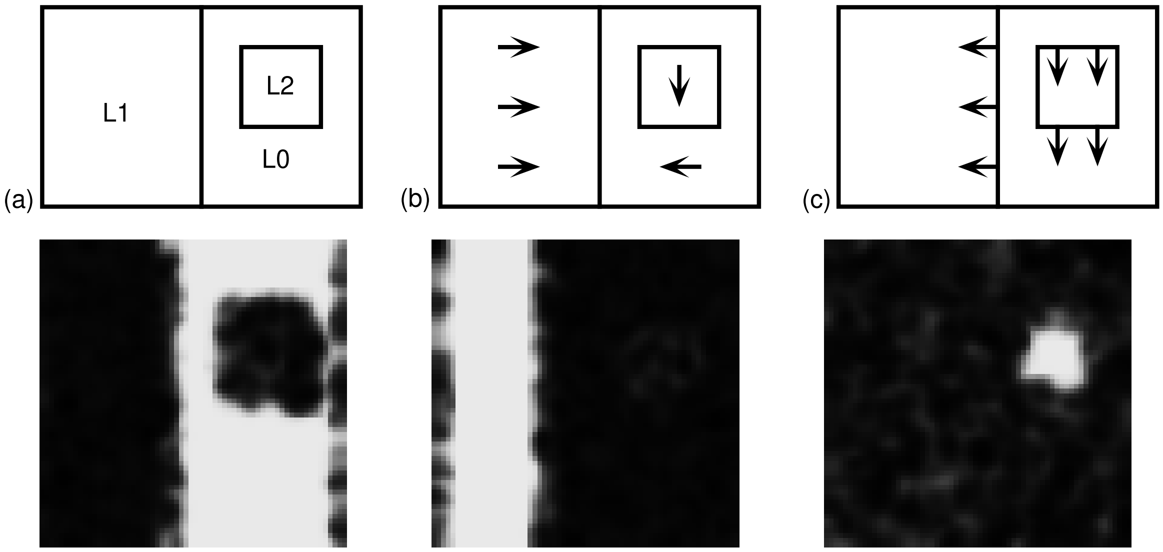


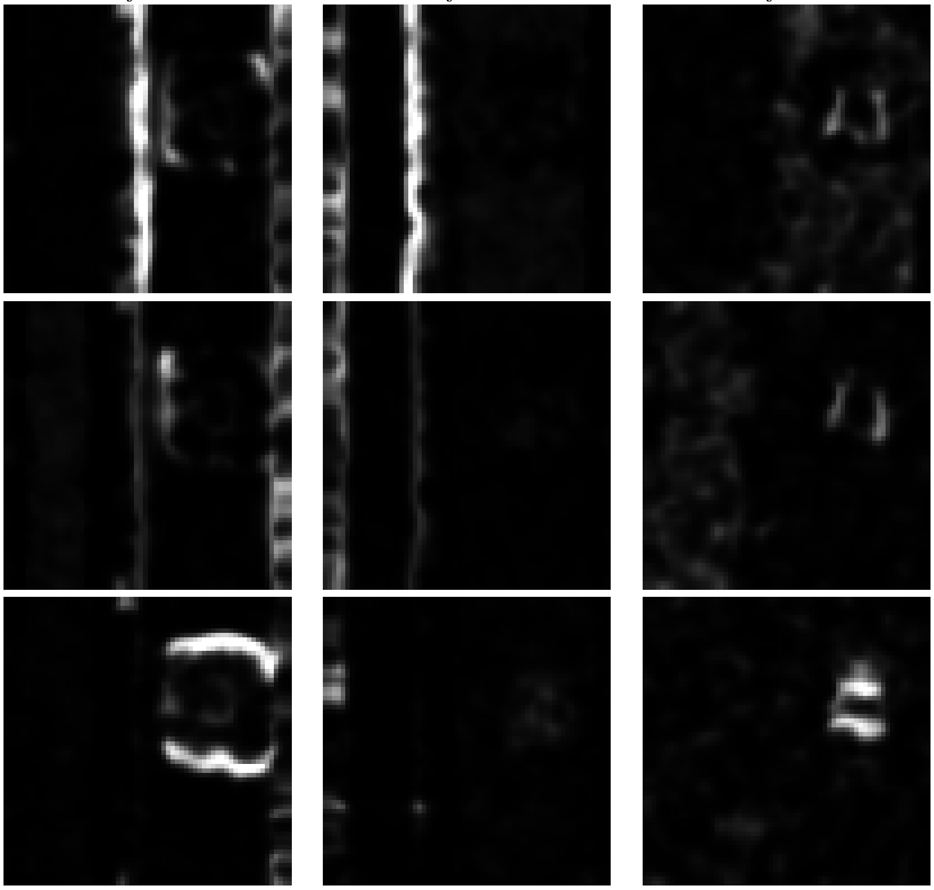
Figure 7: Stages in the recovery of a layer ordering graph for a syntheic image sequence with 3 layers. Description of input input sequence: (a) layer regions in scene, (b) motion in each region, (c) motion of support/boundaries. Layer l_0 is on the right and is moving left, occluding layer l_1, which is on the left moving right. Layer l_2 is a smaller region moving down, and is occluding l_0. The bottom row shows the first-order support layers $s_i(x, y)$.

(o_{ij}) j=0 j=1 j=2

i=0

i=1

i=2



(a)

(b)

	0	1	2
0	0.023	0.024	0.001
1	0.005	0.002	0.001
2	0.019	0.001	0.006

(c)

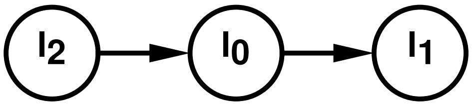


Figure 8: Continuation of previous figure. (a) Computed occlusion maps $o_{ij}(x, y)$ indicating where layer l_i is occluding l_j. (b) Occlusion matrix showing amount of occlusion between pairs of layers. (c) Occlusion graph constructed from occlusion matrix, reflecting the depth ordering of of objects in the scene. A simple linear ordering is found.

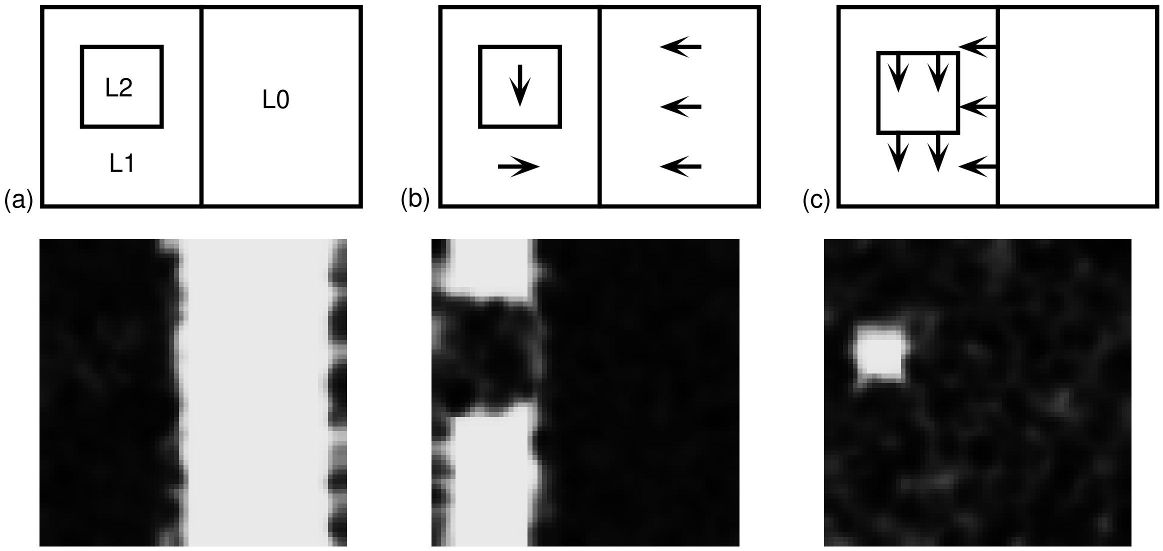


Figure 9: Same as example in previous figures, except layer l_2 is occluding l_1 rather than l_0. (a,b,c) Description of input sequence. The bottom row shows the first-order support layers $s_i(x, y)$.

derivatives need to be used, as will be the case in the following section for computing second-order motion. In addition, layers can impose constraints on the shape of motion support, such as smoothness or morphological constraints, which respect the object boundaries in the scene. Reliable estimation of the shape of first-order support over time is the crux of a second-order method; for this reason a layers approach to second-order motion perception has many advantages over a purely local model in terms of robustness to noise.

Using second-order motion models applied to layers, we devised a method for constructing an ordering graph of motion layers. Second-order motion is used to determine "ownership" of a boundary, which indicates whether the boundary is occluded or not. We further derived a measure for dertermining the occlusion relationship between two layers, by testing whether the support of one motion layer was moving with the predicted local velocity of another motion layer. Our algorithm is implemented upon an abstraction of first-order motion, and can use any motion layer model which provides global hypotheses in the form of velocity fields, and any local mechanism to compute the conditional probabilty of a velocity given a local image region.

The method we have presented here allows machine vision systems to recover a simple representation of depth in the scene, without the expense of a full structure-from-motion or ego-motion computation. This type of "2.1D" model may be suffcient for many practical machine vision tasks which do not require a full depth map of the scene. However, even for those methods that do require full structure-from-motion information, early decomposition into a ordered layers representation can help depth recovery as it provides a set of homogenous regions of data to work with, and knowledge about which objects are closest and farthest from the camera.

References

[1] Adelson E. H., and Anandan, P., "Ordinal characteristics of transparency", in *Proc. AAAI Workshop on Qualitative Vision*, pp. 77-81, Boston, MA, 1990.

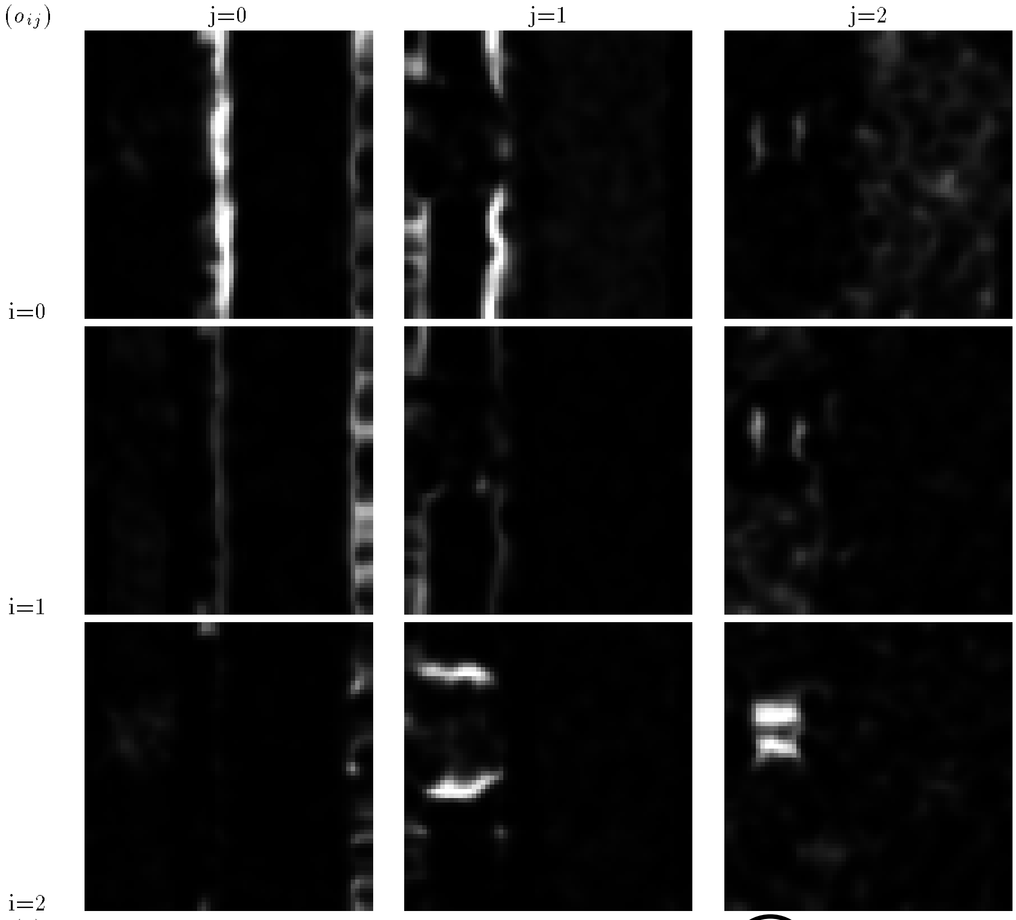
[2] Adelson, E. H., personal communiction.

[3] Black, M. J., and Anandan, P., "The Robust Estimation of Multiple Motions: Affine and Piecewise-Smooth Flow Fields", Xerox PARC Technical Report SPL-93-092, December, 1993.

[4] Black, M. J., and Anandan, P., "A framework for the robust estimation of optical flow", In *Proc. Intl. Conf. Computer Vision, ICCV-93*, pages 231-236, Berlin, Germany, May 1993.

[5] Blake, A. and Zisserman, A. *Visual Reconstruction.* Cambridge, MA: MIT Press. 1987.

[6] Cavanagh, P. and Mather, G. "Motion: The long and Short of it", *Spatial Vision* Vol. 4, Nos. 2/3, pp. 103-129, 1989



(a)

(b)

	0	1	2
0	0.021	0.014	0.001
1	0.001	0.001	0.000
2	0.001	0.009	0.008

(c)

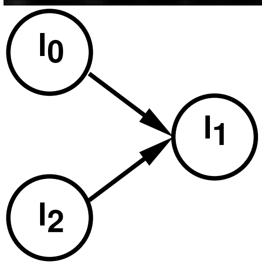


Figure 10: Second-order motion maps for previous figure. (c) Occlusion maps $o_{ij}(x, y)$ indicating where layer l_i is occluding l_j. (d) Occlusion matrix showing amount of occlusion between pairs of layers. (e) Occlusion graph constructed from occlusion matrix, reflecting the depth ordering of of objects in the scene. A tree is recovered, since the relative depth of layers l_0,l_2 is ambiguous.

[7] Darrell, T., and Pentland, P. "Cooperative Robust Estimation of Multiple Layers of Support", to appear, *IEEE Trans. Pattern Anal. and Mach. Intell.*

[8] Darrell, T., and Pentland, A. P., "Robust Estimation of a Multi-Layer Motion Representation", in *Proceedings IEEE Workshop on Visual Motion*, pp. 173-177, 1991.

[9] Darrell, T., and Simoncelli, E. P., "Separation of Transparent Motion into Layers using Velocity-Tuned Mechanisms", presented at *Proceedings Assn. Research in Vision and Opth. annual conference (ARVO93)*, available as MIT Media Lab Perceptual Computing Technical Report TR-244.

[10] Fleet, D.J. and Langley, K., "Computational analysis of non-Fourier motion", *Vision Research* Vol. 34, No. 22, pp. 3057-3079, 1994

[11] Leonardis, A., "Recover and Select Paradigm– A Robust Approach to Estimation of Parametric Models", in *Proceedings NSF/ARPA Wkshp. on Performance vs. Methodology in Computer Vision*, Seattle, June 1994.

[12] Meer, P., Mintz, D., and Rosenfeld, A., "Robust regression methods for computer vision: A review", *Intl. J. Computer Vision*, vol 6, pp. 60-70, 1991.

[13] Meer, P., and Haralick, R., eds, *Proceedings Workshop on Performance vs. Methodology in Computer Vision*, CVPR-94, Seattle, 1994.

[14] Nitzberg, M., and Mumford, D., "The 2.1-D Sketch", *Proc. 3rd Intl. Conf. on Computer Vision*, pp. 138-144, 1990.

[15] Pentland, A., "Automatic recovery of deformable part models", *Intl. J. Computer Vision*, vol 4, pp. 107-126, 1990.

[16] Poggio, T., Torre, V., and Koch, C. "Computational vision and regularization theory", *Nature*, vol. 317(26), 1985.

[17] Irani, M., Rousso, B., and Peleg, S., Computing Occluding and Transparent Motions *Intl. J. Computer Vision*, vol. 12(1), pp. 5-16, 1994.

[18] Terzopoulos, D. "The computation of visible surface representations", *IEEE Trans. Pattern Anal. and Machine Intell.*, vol. 10:4, 1988

[19] Simoncelli, E., Adelson, E. H., and Heeger, D. J., "Probability Distributions of Optical Flow", *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1991.

[20] Wang, J., and Adelson, E. H., "Layered Representations for Image Sequence Coding", *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1993.